HN

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/808,469	03/14/2001	Peter Warnes	ARC.015A	3548

27299 7590 07/05/2005

GAZDZINSKI & ASSOCIATES
11440 WEST BERNARDO COURT, SUITE 375
SAN DIEGO, CA 92127

EXAMINER	
STEELMAN, MARY J	
ART UNIT	**PAPER NUMBER**
2191	

DATE MAILED: 07/05/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	**Application No.** 09/808,469	**Applicant(s)** WARNES, PETER	
	Examiner Mary J. Steelman	**Art Unit** 2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 April 2005*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17 and 19-45* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17 and 19-45* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *18 April 2005* is/are: a)☒ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date *4/18/2005*.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____.

5) □ Notice of Informal Patent Application (PTO-152)

6) ☒ Other: *copy of accepted drawings*.

ND

DETAILED ACTION

1. This action is in response to RCE filed 18 April 2005. Per Applicant's request, claims 1, 8, 14, 19, 20, 25, 26, 34, 36, and 40 have been amended. Claim 18 has been previously canceled. New claims 42-45 have been added. Claims 1-17, 19-45 are pending.

Specification

2. Per Applicant's request, the Specification has been amended.

Drawings

3. In view of the receipt of 'Replacement Sheets' for FIG. 2, FIG. 3, and FIG. 33, the prior objections are hereby withdrawn.

Information Disclosure Statement

4. IDS received 18 April 2005 has been considered.

Claim Rejections - 35 USC § 112

5. In view of Applicant's amendments and comments, the prior 35 USC 112 second paragraph rejections are hereby withdrawn.

6. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. Claims 43-45 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Claim 43 recites the limitation "...based at least in part on the least one customized parameter and at least one prototype description and at least one extension logic description" in lines 26-27 (page 12). There is insufficient antecedent basis for these limitations in the claim.

Claim Objections

8. Claims 40, 42, and 43 are objected to because of the following informalities:

Claim 40 is objected to because the status identifier is incorrect. Claim 40 recites "Previously presented", should be –Currently amended--. Examiner will treat claim 40 as if it stated 'Currently Amended.'

Claim 42 is objected to because it recites "A reduced instruction rest...", should be –A reduced instruction set...-- Change 'rest' to 'set'. For examination purposes, claim 42 will be treated as if it recited 'set.'

Claim 43 is objected to because it recites (page 12, line 26), "...in part on the least one customized parameter...", should be --...in part on the at least one customized parameter...-- For examination purposes, claim 43 will be treated as if it recited '...on the at least one customized parameter...'

9. Appropriate correction is required.

Examiner's Amendment

10. With Applicant's approval (Rob Gazdzinski, Reg. No. 39,990 on 21 June 2005) , the status identifier in the amendments submitted 18 April 2005, for claim 40, is changed to read "Currently Amended."

Claim Rejections - 35 USC § 102

11. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

12. Claims 8-10, 14, 19, and 25 are rejected under 35 U.S.C. 102(e) as being anticipated by US Patent 6,587,939 B1 to Takano.

Per claim 8:

-a RISC processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

Takano: Col. 10, line 11 disclosed a RISC processor core. Col. 5, lines 2-40 and FIG. 5, #12, #13: processor / pipeline with fetch, decode, and execute stages. Col. 5, line 2, 'fetch', line 7, 'decode', line 51, 'execute.'

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

Takano: FIG. 1 and col. 2, 18-20, "storage region", col. 3, lines 37-48 provides a description of the processor (#151) and communication with storage device (instruction cache, #105) to hold a plurality of program instructions. Data interface / data communication is indicated in FIG. 1 with arrows between various components.

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:
Takano disclosed a base instruction (source file, col. 7, line 15) and a compressed instruction set (compressed instruction, col. 7, lines 60-65), optimized by compression.

-determining the static frequency of each of said instruction types from said base instruction set;
Takano disclosed determining the frequency of instruction sequences at col. 7, lines 19-65.

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;
Takano disclosed determining the number and type of instruction sequences at col. 7, lines 19-65 (count frequency of identical instructions, col. 7, line53).

-creating a compressed instruction set encoding to generate said compressed instruction set.
Takano disclosed creating a compressed instruction set at col. 7, line 67-col. 8, line 4.

Per claim 9:

-the act of creating a compressed instruction set comprises selecting "N" instructions having the greatest frequency of occurrence, said selected "N" instructions permitting said program to be compiled with a predetermined size.

Takano provided more details related to selecting instructions with the greatest frequency of occurrence. See FIG. 9 for frequency count of instructions. Col. 9, lines 30-34, "In the case that the requirement of defining compressed instructions is the occurrence frequency of the corresponding sequence or executable instruction is determined as no smaller than 2..." Thus Takano suggested that a specified count (N) is used to determine which instructions are chosen to be a candidate for compression, suitably (col. 2, lines 28-32) reconfiguring/ compacting for storing the target program (program compiled with a predetermined size – to fit).

Per claim 10:

-said optimized instruction set also comprises at least one extension instruction adapted to perform a predetermined function, said processor further comprises an extension logic unit adapted to execute said at least one extension instruction.

Takano disclosed (col. 12, lines 48-52) "that the compressed instructions as defined are extended into the corresponding executable instructions with reference to the compressed/executable instruction correspondence table..."

Per claim 14:

-a reduced instruction set (RISC) processor core having a pipeline comprising at least instruction fetch, decode, and execute stages;

-a memory interface adapted to at least read program instructions from a program memory and provide said instructions to said pipeline;

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set having a predetermined number and type of instructions necessary for correct instruction set execution on said processor core, said predetermined number and type based at least in part on the static frequency of occurrence of instructions within said base instruction set.

(See rejection of limitations as addressed in claim 8 above.)

Per claim 19:

A method of enhancing the performance of a reduced instruction set processor, said processor having a multi-stage instruction pipeline and an instruction set having at least a base instruction set, said base instruction set having a plurality of instruction types associated therewith; comprising;

-providing a program having a plurality of instructions;

-determining the static frequency of each of said instruction types within said plurality of instructions of said program;

-selecting a number instructions having the greatest frequency of occurrence, said selected instructions allowing said program to be compiled with a predetermined size;

-compiling said program based at least in part on said selected instructions.

(See rejection of limitations as addressed in claim 8 above.)

Per claim 25:

A pipelined digital processor (See limitations addressed in claim 8.), comprising:

-reduced instruction set computer (RISC) processor means having an instruction pipeline comprising at least means for instruction fetch, means for instruction decode, and means for instruction execution;

-means for data interface, said means for data interface being in data communication with said processor core, said means for data interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

-optimized instruction means comprising base instruction means and compressed instruction means, said compressed instruction means being generated by the method comprising:

-determining the static frequency of each of said instruction types from said base instruction means;

-determining the number and type of instructions necessary for instruction execution based at least in part on said act of determining the static frequency;

-creating a compressed instruction encoding to generate said compressed instruction means.

(See limitations addressed in claim 8.)

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

13. Claims 1-7 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,408,428 B1 by Schlansker et al., in view of US Patent 6,101,592 to Pechanek et al., and further in view of US Patent 5,587,939 B1 to Takano.

Per claim 1:

A method of optimizing the instruction set of a digital processor…comprising:

Schlansker disclosed a "system as well as methods for automated design" (col. 88, line 23) and a computer readable medium (col. 92, lines 7-8) for development of compressed and optimized instruction sets for processors.

Schlansker failed to disclose:

-having a mixed 16 bit and 32 bit instruction set architecture

However Pechanek disclosed configuring an instruction set including mixed 16 bit and 32 bit instructions (Col. 4, lines 53-65).

Schlansker failed to disclose:

-(i) providing a program having a plurality of different instruction types, including both 16 bit and 32 bit types;

However Pechanek disclosed configuring an instruction set including mixed 16 bit and 32 bit instructions (Col. 4, lines 53-65).

Schlansker /Pechanek failed to disclose:

-(ii) determining the static frequency of each of said instruction types from a base instruction set;

-(iii) determining the number and type of instructions necessary for correct instruction set execution based at least in part on said act of determining the static frequency;

However, Takano disclosed (col. 2, lines 18-27) an "invention to provide an information processing apparatus having a compact storage region required of an executable program with a reduced number of instructions constituting the executable program…characterized by provision of a executable instruction extracting unit for extending a compressed instruction into a plurality of corresponding executable instructions, and a reconfiguration unit for reconfiguring and optimizing the executable instruction extracting unit suitable for the executable program."

See Takano, FIG. 8, #401: the instruction sequence occurrence frequency detecting unit. Col. 7, lines 20-23, "The instruction sequence occurrence frequency detecting unit serves to obtain the occurrence frequency of the instruction sequences contained in the assembler source file (determining the static frequency of each of said instruction types from a base instruction set) …" The instruction frequency is detected and (col. 7, line54-65) "registered in the compressed/executable instruction correspondence table (by type)..." Col. 8, lines 11-14, "...under the condition that the result of execution of the executable program is not influenced by the modification (necessary for correct instruction set execution) ..."

-(iv) creating a compressed instruction set encoding to generate a compressed instruction set based at least in part on said act of determining.

Schlansker disclosed:

(Col. 3, lines 52-54, "The method then programmatically generates a description of the new candidate processor in a hardware description language from the new specification.", col. 4, lines 42-48, "The instruction format designer programmatically generates an instruction format from the datapath specification and the abstract instruction set architecture specification. This instruction format includes instruction templates representing VLIW instructions executable in the VLIW processor, instruction fields of each of the templates, and bit positions and encoding for the instructions fields", col. 14, lines 10-12, "The role of spacewalker is to identify processors (identifies processor designs) which deliver the greatest possible performance at the lowest possible cost.", (compressed / optimized), col. 18, lines 40-46, "The instruction format specifies the instructions capable of being executed in a VLIW processor design. These instructions are represented as instruction templates...also includes the instruction fields within each template, and the bit positions and encodings for the instruction fields", col. 22, lines 20-21, "...the processor design includes the instruction format specification...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker's invention regarding the automatic design of processor systems, by including details disclosed by Pechanek, to identify a mixed 16-bit and 32-bit instruction set architecture used in configuring a processor, because Pechanek recognized (col. 1, lines 42-45) that processors may be developed with new instructions providing optimized capabilities for specific applications. It would have been obvious to further modify Schlansker / Pechanek by including static frequency of instructions when creating a compressed instruction set, as disclosed by Takano, as all inventions are directed towards optimal processor configuration (selection of optimal subset of instructions) for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains."

Per claim 2:

-re-evaluating said compressed instruction set using at least said steps (i), (ii), and (iii); Schlansker disclosed the "instruction format" as a part of the processor design. The candidate processor is evaluated and repeated, using metrics to optimize. Schlansker disclosed (col. 3, lines 65-67), "The process of specifying and evaluating candidate processors may be repeated to explore the parameterized design space.", col. 6, lines 58-59, "This process is iterated until systems of adequate quality are identified.", col. 22, lines 23-26, "One form of optimization used in the system is the customization or optimization of the processor design based on internal usage statistics..."

More specifically, Takano disclosed compressing and extending an instruction set / format, in order to reduce storage requirements. Additionally, Takano disclosed repeating the compression technique as needed at col. 13, lines 21-27.

Regarding the limitation:

-generating an instruction set encoding for said compressed instruction set.

Schlansker disclosed, (col. 22, lines 31-36), "The MDES extraction process programmatically generates an MDES description for driving a retargetable compiler. The MDES is represented in database tables that provide the op code repertoire of the processor, their IO formats, their latencies and resource sharing constraints of the operations…", col. 28, line 66- col. 29, line 2, "The abstract ISA spec is a machine-readable data structure that specifies register files, operation groups, ILP constraints, and architecture parameters." He failed to specifically disclose 'generating an instruction set encoding for said compressed instruction set.'

However, Takano explicitly disclosed (col. 3, lines 51-54), "The executable instruction extracting unit reconfiguration unit serves to reconfigure the executable instruction extracting unit on the basis of the compressed/executable instruction correspondence table." Takano, col. 7, lines 56-65, making use of frequency counts, instructions are compressed and registered in the compressed/executable instruction correspondence table. See FIG. 10. Thus, Takano disclosed generating an encoding for the compressed instruction set.

Schlansker / Pechanek failed to explicitly disclose 'static frequency.'

-regarding “static frequency” as referenced in steps ii and iii of claim 1 above. Takano disclosed this feature as noted in the rejection of claim 1 above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker/ Pechanek regarding the automatic design of processor systems, by including static frequency of instructions when creating/ generating a compressed instruction set, and re-evaluating, as disclosed by Takano, as all inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Schlansker, col. 1, line 55, referenced the desire for “optimizing architectures for particular application domains.”

Per claim 3:

-the act of providing a program comprises providing an assembly language program.
Schlansker disclosed (Col. 16, lines 45-49), “…programs as well as the input and output data structures are implemented in software stored on the workstation’s memory system…may be implemented using standard programming languages…” Schlansker / Pechanek failed to specify “assembly language.” However, Takano disclosed “assembler source file” at col. 7, line 15 and FIG. 9.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker / Pechanek regarding the automatic design of processor systems, by specifically noting that assembly language could be used as the source input because

Schlansker broadly suggested (col. 16, lines 45-49) the source could be a "standard programming language" and Pechanek disclosed (col. 12, lines 24-25) "assembler programming in compacted code is not precluded."

Per claim 4:

-...said instruction types by frequency of usage.

Schlansker disclosed (col. 7, line 41-45), "The re-targetable compiler schedules an application program and generates a number of statistics...The operation issue statistics provide histograms indicating the static and dynamic opcodes usage...(sort by frequency)", broadly disclosing that statistics are collected. Schlansker / Pechanek failed to specify 'sorting'. However, Takano more explicitly disclosed the sort. See FIG. 9 which shows a count per instruction.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker / Pechanek regarding the automatic design of processor systems, by including more details related to frequency counts of instructions when creating/ generating a compressed instruction set as both inventions are directed towards optimal processor configuration for a target program, resulting in reduced storage requirements (Takano, col. 2, lines 29-31.) Choosing the most frequently executed instructions as compressed instructions optimizes the configurable processor. Schlansker, col. 1, line 55, referenced the desire for "optimizing architectures for particular application domains." Likewise Pechanek suggested (col. 11, line 64-col. 12, line 12) that highly sequential code segments are chosen for compaction.

Per claim 5:

-digital processor includes an extension logic unit adapted to execute at least one extension instruction, and the act of providing comprises providing a program having said at least one extension instruction, said at least one execution instruction being executable by said extension logic unit.

Schlansker disclosed, (col. 23, lines 2-22), "In the first case, the datapath specification may have been specified by hand...In the second case, the concrete ISA specification may be provided as input based on some existing processor design...Alternatively, the developer may wish to optimize an existing concrete ISA specification for a particular application or application program (extension logic). To support these design scenarios, the system includes modules for extracting an abstract ISA specification...and concrete ISA specification respectively...may alter the abstract ISA specification...One particular example is the use of custom templates based on operation issue statistics...may alter the opcodes repertoire and ILP constraints to achieve an optimized design based on cost/performance trade-offs." Also see col. 3, lines 32-34, "A compiler, re-targeted to the candidate processor, generates operation issue statistics for an application program to be executed in the candidate processor.")

Per claim 6:

-the act of creating a compressed instruction set comprises selecting "N" instructions having the greatest frequency of occurrence, said selected "N" instructions permitting said program to be compiled with a predetermined size.

Schlansker disclosed giving consideration to code size. (Col. 83, line 54-57) "The compiled application is assembled and linked to determine the application's code size (and ROM area) as well as an estimate of the number of cycles needed to execute it…" Pechanek disclosed selecting instructions that are highly sequential (col. 11, line 65-col. 12, line 12).

Takano provided more details related to selecting instructions with the greatest frequency of occurrence. See FIG. 9 for frequency count of instructions. Col. 9, lines 30-34, "In the case that the requirement of defining compressed instructions is the occurrence frequency of the corresponding sequence or executable instruction is determined as no smaller than 2…" Thus Takano suggested that a specified count (N) is used to determine which instructions are chosen to be a candidate for compression.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Schlansker / Pechanek regarding the automatic design of processor systems, by including more details related to executable extension logic when creating/ generating a compressed instruction set. Selecting the most frequent occurrences of instructions to compress, as disclosed by Takano contributes to optimal design. All referenced inventions are directed towards optimization. Schlansker (col. 1, line 55) referenced the desire for "optimizing architectures for particular application domains."

Per claim 7:

-the act of determining a compression ratio for said compressed instruction set, said compression ratio being related to the ratio of the number of compressed instructions to the total number of original instructions.

Schlankser disclosed (col. 7, line 57 – col. 8, line 3), "The system includes a program or programs that implement search heuristics to select candidate processor designs for evaluation. These search heuristics use information about a candidate processor's cost and performance to select other candidates. A performance evaluator computes the performance of a candidate processor in terms of execution cycles. A cost evaluator evaluates the cost of a candidate processor based on costs of individual components in the hardware description, which lists instances of macrocells and their corresponding areas, power consumption, etc...internal resource usage information (compression ration) may be used to refine or focus the search more effectively.")

Schlansker did not explicitly disclose a "compression ratio". However, he did disclose that heuristics and internal resource usage information are involved in the decision for an optimal design. Pechanek suggested consideration for selection based on highly sequential sections of code, Huffman encoding / decoding functions. Takano disclosed consideration to selecting code segments with the highest frequency occurrence (col. 7, lines 7-14) to produce an optimal sized (ratio of compressed to original) instruction set.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Schlansker / Pechanek / Takano, to broadly interpret the consideration of heuristics and internal resource usage information as including the ratio of compressed instructions to original instructions, as all heuristics are meant to optimize processor instruction set design.

12. Claims 11-13, 15-17, 32-34, and 42-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 5,587,939 B1 to Takano, and further in view of US Patent 6,477,683 B1 to Killian.

Per claims 11-13:

Takano suggested that the encoding may provide a plurality of instruction slots (col. 10, lines 21-26), where he disclosed that the bit width may be designed shorter, thus suggesting a plurality of instruction slots. See FIG. 17 which shows an opcode and bits encoding an instruction. Takano failed to disclose encoding of opcodes, and instruction slots.

Regarding the limitations:

-an encoding structure having an opcode and a plurality of instruction slots.

-plurality of instruction slots comprise two slots, each of said slots having two 14-bit instructions / encoding structure comprises 32-bits, and said opcode is disposed within the last four bits thereof.

Killian suggested (col. 17, lines 27-35) a user enters configuration and extension options…parameters set by the user…in defining the core processor, (col. 20, lines 31-34), a user limits bit sizes of data types / 12 bit or 20 bit or any other size integers, and (col. 25, lines 35-41) benchmark programs are used and estimates of different architecture design decisions may be varied by a user in search of an optimal solution.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to instruction encoding in processor design as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), Takano suggested that encoding may be shorter for compressed instructions, whereas Killian suggested that a designer defines encoding, both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 15:

-an encoding structure having an opcode and a plurality of instruction slots.

(See limitations addressed in claim 11.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to instruction encoding in processor design as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), Takano suggested that encoding may be shorter for compressed instructions, whereas Killian suggested that a designer defines encoding, both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 16:

-said instruction set includes at least one extension instruction, said at least one extension instruction adapted to perform a predetermined function upon execution within said processor. Takano explicitly disclosed extension logic (col. 5, lines 8-9), "...in the case of a compressed instruction, it is extended into the corresponding executable instructions..." See FIG. 2 which shows a table of a compressed / executable instruction correspondence table, depicting the predetermined function to be performed.

Per claim 17:

-an extension logic unit adapted to execute said at least one extension instruction.

Takano (col. 5, lines 14-25) disclosed an executable instruction extracting unit which extend a compressed instruction (extension instruction) into the corresponding executable instruction (adapted to execute).

Per claim 32:

A user-configured and extended pipelined RISC processor, comprising:

-a processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

Takano disclosed a RISC type processor (col. 10, line 11). Takano disclosed fetch, decode, execute pipeline (col. 5, lines 1-41)

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

Takano disclosed (FIG. 1) main memory and instruction cache inherently interfaced with a processor (col. 3, lines 37-48).

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

Takano disclosed base and compressed instruction sets. See rejection of claim 25 above.

-determining the static frequency of each of said instruction types from said base instruction set;

Takano disclosed consideration given to static frequency. See rejection of claim 25 above.

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

See rejection of claim 25 above.

Regarding the limitation:

-creating a compressed instruction set encoding to generate said compressed instruction set, said creating comprising using an encoding structure having a opcodes and two instruction slots, each of said slots having two 14-bit instructions, and selecting two instructions having the greatest frequency of occurrence, said selected two instructions permitting said program to be compiled with a predetermined size.

Takano disclosed creating a compressed instruction set. Takano failed to disclose specific details related to the arrangement of opcodes and instruction slots. However, Killian disclosed various bit sized instructions. Killian disclosed that a user enters various design constraints (col. 9, lines 55-61) See examples of options available (16 / 32 bits) at col. 11, lines 30- col. 12, line 31.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to instruction encoding in processor design as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), Takano suggested that encoding may be shorter for compressed instructions, whereas Killian suggested that a designer defines encoding, both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 33:

-said structure comprises a 32-bit encoding structure with said opcodes disposed within the last four bits thereof.

See rejection of limitations as addressed in claim 13 above. It is obvious that a designer may designate a location, such as the last four bits, for a specific encoding structure.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to instruction encoding in processor design, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), Takano suggested that encoding may be shorter for compressed instructions, whereas Killian suggested that a designer defines encoding, both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 34:

-A user-configured and extended pipelined RISC processor, comprising:
(See rejection of limitations as addressed in claim 32 above.)

-a processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;
(See rejection of limitations as addressed in claim 32 above.)

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

(See rejection of limitations as addressed in claim 32 above.)

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

(See rejection of limitations as addressed in claim 32 above.)

-determining the static frequency of each of said instruction types from said base instruction set;

(See rejection of limitations as addressed in claim 32 above.)

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

(See rejection of limitations as addressed in claim 32 above.)

-creating a compressed instruction set encoding to generate said compressed instruction set, said creating comprising using a 32-bit encoding structure having an opcodes and a plurality of instruction slots, said opcodes being disposed within at least the last 4-bits of said structure, and selecting a corresponding number of instruction having the greatest frequency of occurrence, said selected instruction permitting said program to be compiled with a predetermined size;

(See rejection of limitations as addressed in claims 9,13, and 33 above.)

Takano failed to disclose:

-wherein both said user-extension and configuration of said processor are performed as part of generating a of a description language model of said processor.

However, Killian disclosed details related to generating a description language model (col. 10, lines 6-21 & col. 21, lines 4-22) using HDL, such as Verilog.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to generating processor features, as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested that compressed / extended instructions for the purpose of optimal processor design, Killian further suggested that the design is modeled in a hardware description language. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 42:

A reduced instruction set (RISC) pipelined digital processor, comprising:

-a user-configured and user-extended RISC processor core having an instruction pipeline comprising at least instruction fetch, decode, and execute stages;

-a data interface in data communication with said processor core, said interface adapted for data communication with a storage device configured to hold a plurality of program instructions;

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set being generated by the method comprising:

-determining the static frequency of each of said instruction types from said base instruction set:

-determining the number and type of instructions necessary for instruction set execution based at least in part on said act of determining the static frequency;

-creating a compressed instruction set encoding to generate said compressed instruction set;

-wherein both said user-extension and configuration of said processor core are performed as part of generating of a description language model of said processor core.

See rejection of claim 32-34 above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to generating processor features, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested that compressed / extended instructions for the purpose of optimal processor design, Killian further suggested that the design is modeled in a hardware description language. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 43:

Takano failed to disclose:

-said generating of a description language model of said processor core comprises:

However, Killian disclosed description language models (HDL) (col. 21, lines 4-22).

-receiving one or more inputs from a user for at least one customized parameter of the processor core;

Killian disclose user input related to parameters (col. 9, lines 55-61).

-generating through an automated process said description language model based at least in part on the at least one customized parameter and at least one prototype description and at least one extension logic description.

Killian (col. 33, lines 4-11) disclosed synthesis using scripts to guarantee functionally correct implementation with optimal performance. Commands in the script read all RTL files relevant to the specific processor configuration. As an example (col. 40, line 37-col. 41, line 30), Killian defines a new opcodes / instruction (SAD-Sum of Absolute Difference). A new instruction class containing the new instruction is defined. A description using a subset of Verilog HDL language for description the combination logic is given. This description is the language model based on a customized parameter, prototype description and extension logic description.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to generating processor features, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested that compressed / extended instructions for the purpose of optimal processor design, Killian further suggested that the design is modeled in a hardware description language, as a result of user input. Both inventions are customizing /

optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 44:

Takano failed to disclose:

-generating of a description language model of said processor core further comprises generating, via an automated process, test code associated with said model.

However Killian disclosed generating test code (col. 29, lines 47-65, col. 31, lines 50-61, col. 32, lines 11-67) using a debugger / simulator / emulator.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to testing processor features, as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further suggested that the design is tested to ensure correct functionality. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

Per claim 45:

Takano failed to disclose:

-description language model includes both functional and structural description language descriptions for the processor core.

However, Killian disclosed description language model. See claim 43 above. See col. 21, lines 4-22, a preprocessor (TPP) scans and constructs a program. The resultant program is executed to produce source code, Verilog (description language model) for describing the detailed processor logic (functional and structural)...

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to processor modeling, as disclosed by Killian because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further provided details related to description language modeling of the optimal processor design. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20).

14. Claims 20-24, 26-31 and 35-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 5,587,939 B1 to Takano, and further in view of US Patent 6,477,683 B1 to Killian, and further in view of US Patent 6,101,592 to Pechanek et al.

Per claim 20:

An application specific integrated circuit (ASIC) comprising:

-a first processor core, said processor core having a pipeline with at least instruction fetch, decode, and execute stages associated therewith;

Takano (col. 5, line 2) 'fetch', (col. 5, line 7) 'decode', (col. 5, line 51) 'execute'.

Takano failed to disclose an application specific integrated circuit. However Killian disclosed an ASIC (col. 11, line 34, col. 32, line 19) suitable for configuring with an optimal instruction set.

-an optimized instruction set comprising a base instruction set and a compressed instruction set, said compressed instruction set having a predetermined number and type of instructions, said predetermined number and type based at least in part on the static frequency of occurrence of instructions within one extension instruction adapted to perform at least one specific operation;

Takano disclosed (col. 5, lines 3-4) a combination of base instructions (executable instruction) and compressed instruction set (compressed instruction). Takano disclosed 'frequency' of occurrence as a factor in compressing instructions (col. 7, lines 19-23).

-at least one storage device adapted to store a plurality of data bytes therein, said at least one storage device being accessible by said first processor core;

Takano disclosed (FIG. 1 – main memory #101, instruction cache, #105) storage device, storing a plurality of data bytes, accessible by a first processor core.

-at least one extension logic unit adapted to facilitate execution of said at least one execution instruction;

Takano explicitly disclosed extension logic (col. 5, lines 8-9), "...in the case of a compressed instruction, it is extended into the corresponding executable instructions..."

Takano suggested various bit widths at col. 10, line 24 but failed to specifically disclose:

-wherein said processor core is adapted to execute both 16 bit and 32 bit instructions...

However, Killian suggested (col. 17, lines 27-35) a user enters configuration and extension options...parameters set by the user...in defining the core processor, (col. 20, lines 31-34), a user limits bit sizes of data types / 12 bit or 20 bit or any other size integers (16 / 32), and (col. 25, lines 35-41) benchmark programs are used and estimates of different architecture design decisions may be varied by a user in search of an optimal solution.

Takano / Killian failed to disclose mixed instruction set execution 'without processor mode switching.'

However, Pechanek disclosed (col. 11, lines 26-62) a translation mechanism whereby a 16-bit half word is translated to a 32 bit function. See FIG. 4A & 4C – 15 bit compacted instruction translated to 32 bit instruction. Processor mode switching is not required.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to instruction encoding in processor design, including ASICs, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), Takano suggested that encoding may be shorter for compressed instructions, whereas Killian suggested that a designer defines encoding, both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). It would have been obvious to further modify Takano/Killian, using Pechanek's translation process because it effectively allows for packing two 16-bit instructions into a word, and efficiently translating into a 32 bit instruction.

Per claim 21:

Takano failed to disclose:

-a second processor core, said second processor core being disposed on the same die as said first processor core.

However, Killian disclosed a DSP (second processor core) may optionally be added, col. 12, lines 33-37.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to an additional processor core, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tense processor generator/RISC), both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). The addition of a DSP further enhances design capabilities.

Per claim 22:

Takano failed to disclose:

-said second processor core comprises a digital signal processor (DSP), said DSP being adapted to perform at least one operation on data provided to said ASIC.

However, Killian inherently disclosed an enhanced DSP (digital signal processing) provided as an optional functional unit (Killian, col. 12, line 37), which performs at least one operation (signal processing) on data.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to an additional processor core, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). The addition of a DSP further enhances design capabilities, useful for a specific application.

Per claim 23:

Takano failed to disclose:

-said DSP is adapted for initiation by an instruction from said first processor core.

However, Killian disclosed (col. 12, lines 33-37) "...the system may provide options for adding (initiation by an instruction from said first processor core) other functional units such as...enhanced DSP..."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to an additional processor core, a DSP, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). The addition of a DSP further enhances design capabilities.

Per claim 24:

Takano failed to disclose:

-at least a portion of the operation of said DSP is controlled by extension registers associated with said first processor core.

However, Killian (col. 27, lines 17-40) suggested the coprocessor (DSP) is controlled by extension registers associated with said first processor core, when regions of register spill code are found, the code is automatically changed (change code execution to used extension registers in coprocessor / DSP) to use the coprocessor instructions and registers (col. 27, lines 29-30).

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to an additional processor core, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), both are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). The addition of a DSP further enhances design capabilities.

Per claim 26:

A method of operating an extended pipelined digital processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set, the method comprising:

(See limitations addressed in claim 20.)

-providing a base instruction set having a plurality of 16 bit and 32 bit instructions;

(See limitations in claim 20.)

-providing an extension instruction set…

Takano explicitly disclosed extension logic (col. 5, lines 8-9), "…in the case of a compressed instruction, it is extended into the corresponding executable instructions…"

Takano failed to disclose "… determined at least in part by selections made by a user…"

However, Killian disclosed a user contributing (col. 9, lines 58-61) to extensibility options chosen.

-providing a compressed instruction set derived at least in part from said base and extension instruction sets;

(See limitations in claim 16.)

Regarding the following limitations, Takano / Killian failed to disclose:

-assigning one of a plurality of predetermined values to at least one bit within a status register within said processor;

-executing at least one instruction from said base instruction set within said pipeline based on a first predetermined value present in said status register;

-executing at least one instruction from said compressed instruction set within said pipeline based on a second predetermined value present in said status register.

However, Pechanek disclosed the development of new instructions specific to a processor (col. 1, lines 42-45). Pechanek further disclosed (col. 14, lines 41-65) a state bit in a register (bit within a status register) that indicates an instruction slot is 'available for execution' (a first predetermined value) or it is 'not available for execution' (a second predetermined value). According to the value of the bit, execution will be determined.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to processor instruction encoding, instruction set bit size, and user input options, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further provided details related to encoding the instructions. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). It would have been obvious to further modify the Takano / Killian invention, using details as disclosed by Pechanek regarding a status bit, related to execution commands to correctly direct / synchronize instruction execution. All references are directed to generating optimal instructions for a processor.

Per claim 27:

Takano / Killian failed to disclose:

-the act of assigning comprises assigning a '1' or '0' value to a low address (L) bit within said register.

However, Pechanek disclosed the development of new instructions specific to a processor (col. 1, lines 42-45). Pechanek further disclosed (col. 14, lines 41-65) a state bit in a register (bit within a status register) that indicates an instruction slot is 'available for execution' (a first predetermined value) or it is 'not available for execution' (a second predetermined value). According to the value of the bit, execution will be determined. Pechanek disclosed (col. 14, line63-64) the appropriate d-bit ...is loaded into bit 31 of that slot. It is inherent that encoding the bit location of an instruction may be selectively determined in the design.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to processor instruction encoding, instruction set bit size, and user input options, as disclosed by Killian, because both references are suggestive of RISC processors

(Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further provided details related to encoding the instructions. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). It would have been obvious to further modify the Takano / Killian invention using details as disclosed by Pechanek regarding a status bit. The location of the bit is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 28:

Takano / Killian / Pechanek failed to specifically disclose:

-the act of providing a compressed instruction set comprises encoding at least a portion of the instruction operation codes (opcodes) for the compressed instructions within a predetermined number of the most significant bits of an instruction word.

However, it is inherent that when designing an instruction set, a user may optionally determine the encoding of opcodes, choosing which bits within an instruction word to place the opcodes. It is a design choice.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for the opcodes to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 29:

Takano / Killian / Pechanek failed to specifically disclose:

-the act of providing a compressed instruction set further comprises encoding a plurality of said compressed instruction with source register fields located in a predetermined relationship to one another within said compressed instructions.

However, it is inherent that when designing an instruction set, a user may optionally determine the encoding of opcodes, choosing which bits within an instruction word to place the opcodes. It is a design choice. Pechanek did disclose (FIG. 8) source register fields Rx, Ry.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for the source fields to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 30:

Takano / Killian / Pechanek failed to specifically disclose:

-the act of encoding with said predetermined relationship comprises encoding the source register fields for respective ones of said plurality of compressed instructions at identical locations.

However, it is inherent that when designing an instruction set, a user may optionally determine the encoding of source register fields, choosing which bits within an instruction word to place the opcodes. It is a design choice. Killian disclosed instruction fields (col. 14, line 46- col. 15, line 37, but failed to disclose placement within the instruction word. Pechanek did disclose (FIG. 8) register fields Rx, Ry, presumably source register fields.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting identical locations for the source fields to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 31:

Takano / Killian / Pechanek failed to specifically disclose:

-the act of providing a compressed instruction set comprises encoding all of the immediate data fields such that they start from the least significant bit (LSB).

However, it is inherent that when designing an instruction set, a user may optionally determine the encoding of source register fields, choosing which bits within an instruction word to place the opcodes. It is a design choice. Killian disclosed instruction fields (col. 14, line 46- col. 15, line 37, but failed to disclose placement within the instruction word. Pechanek did disclose (FIG. 8) register fields Rx, Ry, presumably source register fields.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting a location (LSB) for immediate data fields to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 35:

A method of operating a user-extended and configured RISC processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set, the method comprising:

(See rejection of limitations as addressed in claim 20 above.)

-providing a base instruction set having a plurality of instructions and an extension instruction set having at least one extension instruction;

(See rejection of limitations as addressed in claim 20 above.)

-providing a compressed instruction set derived at least in part from said base instruction set;

(See rejection of limitations as addressed in claim 20 above.)

-assigning a value to a compressed instruction selection bit within a status register within said processor;

(See rejection of limitations as addressed in claims 25 & 26 above.)

-executing at least one instruction from said base instruction set within said pipeline based on a first predetermined value present in said status register;

(See rejection of limitations as addressed in claims 25 & 26 above.)

-executing at least one instruction from said compressed instruction set within said pipeline based on said assigned value of said compressed instruction selection bit in said status register.

(See rejection of limitations as addressed in claim 25 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to processor instruction encoding, instruction set bit size, and user input options, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further provided details related to encoding the instructions. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). It would have been obvious to further modify the Takano / Killian invention using details as disclosed by Pechanek regarding a status bit as it is a manner of directing the execution command when synchronizing instruction execution. All references are directed to generating optimal instructions for a processor.

Per claim 36:

-the act of providing a compressed instruction set comprises encoding at least a portion of the instruction operation codes (opcodes) for the compressed instructions within a predetermined number of the most significant bits of an instruction word.

(See rejection of limitations as addressed in claim 28 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for the opcodes to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 37:

-the act of providing a compressed instruction set further comprises encoding a plurality of said compressed instructions with source register fields located in a predetermined relationship to one another.

(See rejection of limitations as addressed in claim 29 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for source register fields to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 38:

-the act of encoding with said predetermined relationship comprises encoding the source register files for respective ones of said plurality of compressed instructions at identical locations.

(See rejection of limitations as addressed in claim 30 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for source register files to be encoded. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 39:

-the act of providing a compressed instruction set comprises encoding all of the immediate data fields such that they start from the least significant bit (LSB).

(See rejection of limitations as addressed in claim 31 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for the immediate data fields to be encoded within the instruction word. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Per claim 40:

A method of operating an extended pipelined digital RISC processor having an instruction pipeline comprising at least instruction fetch, decode, and execute stages, a storage device configured to hold a plurality of program instructions, and an optimized instruction set comprising both 16-bit and 32-bit instructions, the method comprising:
(See rejection of limitations as addressed in claim 8 & 35 above.)

-providing an extension instruction set having a plurality of user-selected extension instructions;
(See rejection of limitations as addressed in claims 8, 10, and 35 above.)

-providing a compressed instruction set derived at least in part from said extension instruction set;
(See rejection of limitations as addressed in claims 8, 20 and 35 above.)

-encoding a plurality of compressed instructions from said compressed instruction set into an instruction word having an op-code.

(See rejection of limitations as addressed in claims 11, 32 & 33 above.)

-assigning one of a plurality of predetermined values to at least one bit within a status register with said processor;

(See rejection of limitations as addressed in claim 27 above.)

-executing at least one of said compressed instructions from said instruction word within said pipeline based on a second predetermined value present in said status register.

(See rejection of limitations as addressed in claim 26 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano's invention for optimizing instructions for a processor, by including some of the details related to processor instruction encoding, instruction set bit size, and user input options, as disclosed by Killian, because both references are suggestive of RISC processors (Takano, col. 10, line 11 & Killian, FIG. 4, Tensa processor generator/RISC), where Takano suggested generating an optimal processor, Killian further provided details related to encoding the instructions. Both inventions are customizing / optimizing processor cores for the purpose of enhancing the design process (Killian, col. 6, line 20). It would have been obvious to further modify the Takano / Killian invention using details as disclosed by Pechanek regarding a status bit as it is a manner of directing the execution command when synchronizing instruction execution. All references are directed to generating optimal instructions for a processor.

Per claim 41:

-said act of encoding comprises encoding two 14-bit compressed instructions into a 32-bit aligned instruction word having said opcodes disposed within at least the last four bits thereof. (See rejection of limitations as addressed in claim 32 above.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Takano / Killian / Pechanek which disclose configuring a processor with an optimal instruction set by selecting the location for the opcodes to be encoded within the instruction word. Two 14 bit compressed instructions encoded into a 32 bit instruction word is known in the art as half words / packing. The location of the selected bits is a design choice and inherently different locations may be chosen. All references are directed to generating optimal instructions for a processor.

Response to Arguments

15. Applicant's arguments with respect to claims 1-17 and 19-31 have been considered but are moot in view of the new ground(s) of rejection.

Applicant has argued, in substance, the following:

(A) Regarding claims 1, 26, and 40, as Applicant has noted on page 15, 2nd and 7th paragraphs and page 16, 3rd paragraph, cited art does not teach "mixed 16-bit and 32-bit instruction set architecture.

Examiner's Response: These are newly added limitations. See rejection of claim 1 above. As an example, Pechanek disclosed mixed 16-bit and 32-bit instructions at col. 4, lines 58-59.

(B) Regarding claim 20, as Applicant has noted on page 15, 5th paragraph, cited art does not teach "...processor core being adapted to execute both 16-bit and 32-bit instructions without processor mode switching."

Examiner's Response: These are newly added limitations. See rejection of claim 20 above. Pechanek disclosed (col. 11, lines 26-62) a translation mechanism whereby a 16-bit half word is translated to a 32 bit function. See FIG. 4A & 4C – 15 bit compacted instruction translated to 32 bit instruction. Processor mode switching is not required.

(C) Regarding claims 8, 14, 19, 25, 32, 34, 35, and 40 , as Applicant has noted on page 16, 5th paragraph, cited art teaches away from a RISC instruction set.

Examiner's Response: These are newly added limitations. See rejections of claims above. New art has been added reciting RISC instruction sets.

(D) Regarding claim 34, as Applicant has noted on page 18, 2nd paragraph, cited art does not teach "user-extension and configuration of the processor are performed as part of generating of a description language model of the processor."

Examiner's Response: These are newly added limitations. See rejections of claims above. Killian disclosed HDL (generate a description language model of the processor) and Verilog at col. 21, lines 4-22 & col. 23, lines 9-48.

Conclusion

16. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached at (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman
06/26/2005

REPLACEMENT SHEET

Accepted by Examiner MS 6·20·05

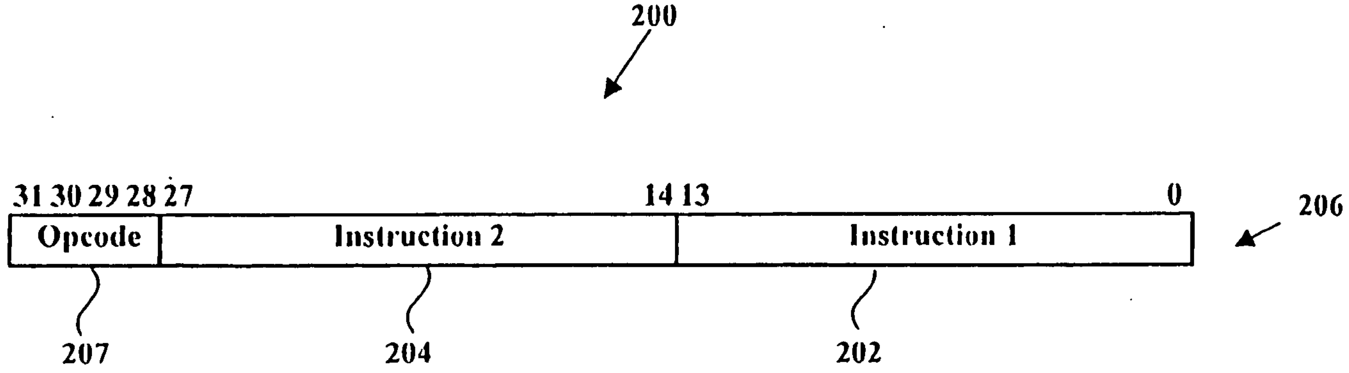

FIG. 2

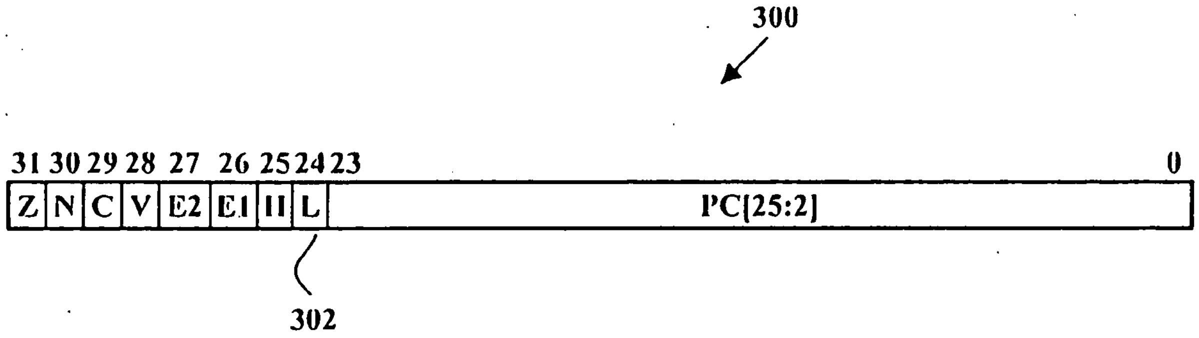

FIG. 3

REPLACEMENT SHEET

Accepted by Examiner 6.20.05

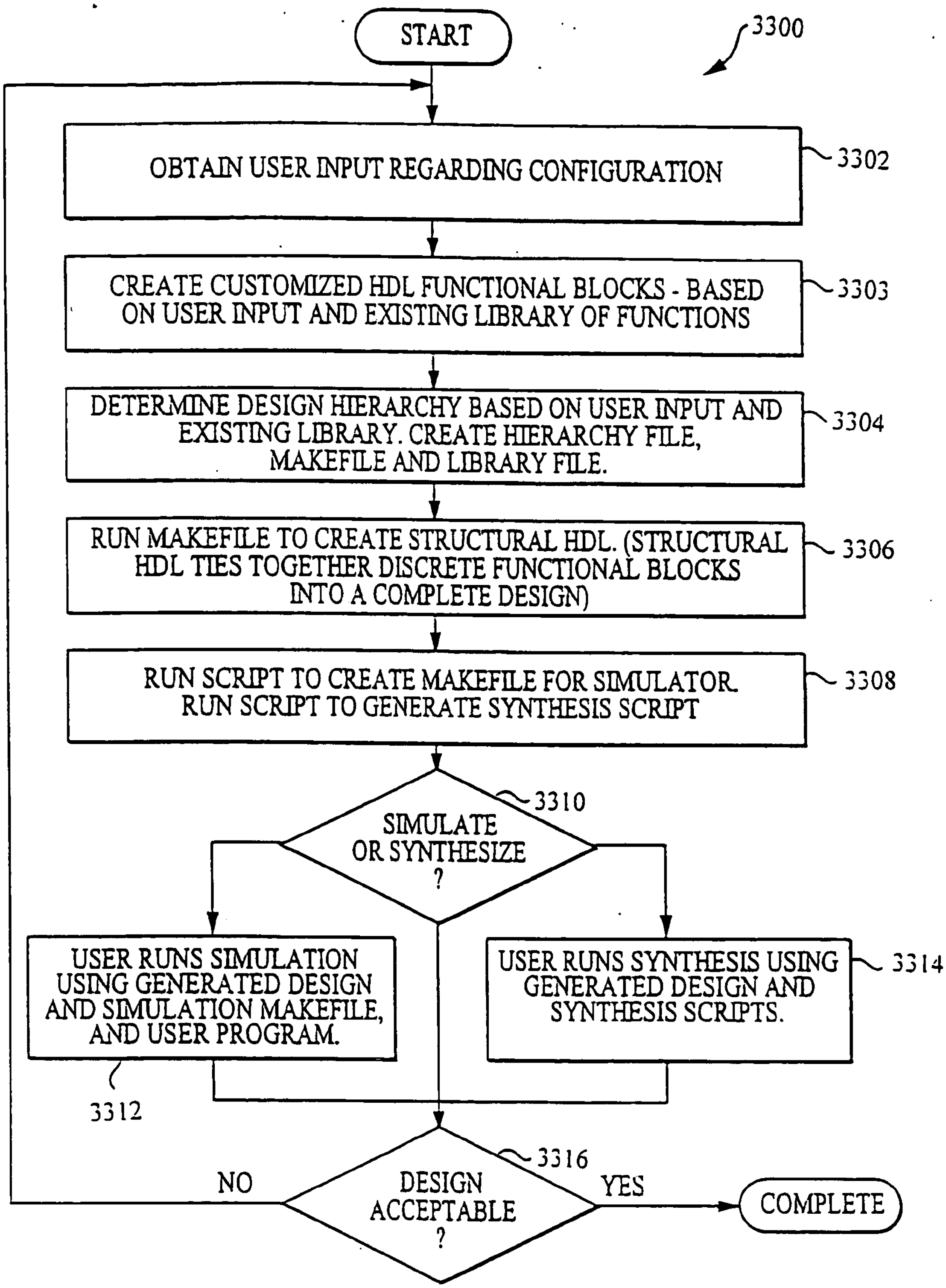
START
3300
OBTAIN USER INPUT REGARDING CONFIGURATION
3302
CREATE CUSTOMIZED HDL FUNCTIONAL BLOCKS - BASED ON USER INPUT AND EXISTING LIBRARY OF FUNCTIONS
3303
DETERMINE DESIGN HIERARCHY BASED ON USER INPUT AND EXISTING LIBRARY. CREATE HIERARCHY FILE, MAKEFILE AND LIBRARY FILE.
3304
RUN MAKEFILE TO CREATE STRUCTURAL HDL. (STRUCTURAL HDL TIES TOGETHER DISCRETE FUNCTIONAL BLOCKS INTO A COMPLETE DESIGN)
3306
RUN SCRIPT TO CREATE MAKEFILE FOR SIMULATOR. RUN SCRIPT TO GENERATE SYNTHESIS SCRIPT
3308
SIMULATE OR SYNTHESIZE ?
3310
USER RUNS SIMULATION USING GENERATED DESIGN AND SIMULATION MAKEFILE, AND USER PROGRAM.
3312
USER RUNS SYNTHESIS USING GENERATED DESIGN AND SYNTHESIS SCRIPTS.
3314
DESIGN ACCEPTABLE ?
3316
NO
YES
COMPLETE

FIG. 33